1. (currently amended) A uniform method for implementing multiple tunneling protocols in a switch or router having a plurality of input interfaces and a plurality of output interfaces, comprising:

      a) providing a finite set of tunnel interfaces, each tunnel interface characterized by a set of tunnel specific attributes, a first of said tunnel interfaces being associated with one tunneling protocol and a second of said tunnel interfaces being associated with a second tunneling protocol different from said first tunneling protocol;

      b) mapping one of the input interfaces to one of said tunnel interfaces; and

      c) mapping said one of said tunnel interfaces to one of the output interfaces, whereby

      multiple tunneling protocols are implemented in a uniform way.

2. (currently amended) The method according to claim 1, wherein:

      said tunnel specific attributes ~~including~~ include parameters identifying tunnel end points.

3. (original) The method according to claim 1, wherein:

      said step of mapping one of the input interfaces to one of said tunnel interfaces is performed by using context data in an arriving packet as a first search key to a first database.

4. (original) The method according to claim 3, wherein:

said arriving packet has a header and said context data is obtained from said header.

5. (original) The method according to claim 4, further comprising:

d) processing said header with said one of said tunnel interfaces to obtain a new header, wherein said step of mapping said one of said tunnel interfaces to one of the output interfaces is performed by using the new header as a second search key to a second database.

6. (original) The method according to claim 1, wherein:

both said step of mapping one of the input interfaces to one of said tunnel interfaces and said step of mapping said one of said tunnel interfaces to one of the output interfaces are performed by using context data in an arriving packet as a first search key to a first database.

7. (original) The method according to claim 6, wherein:

said arriving packet has a header and said context data is obtained from said header.

8. (previously presented) The method according to claim 5, wherein:

the one of the output interfaces is one of an L2 (layer two) and an L3 (layer three) interface, and said step of using the new header as a second search key to a second database yields one of an L2 and an L3 interface.

9. (previously presented) A uniform method for implementing multiple tunneling protocols in a switch or router, comprising:

a) associating an input interface, an output interface, and an information database with each of said multiple tunneling protocols;

b) associating a mapping interface and a mapping information base with each of said multiple tunneling protocols; and

c) uniformly implementing a tunneling protocol by selecting an input interface, an output interface, and an information database associated with the tunneling protocol to be implemented.

10. (cancel)

11. (withdrawn) The method according to claim 9, wherein:

for IP over IP origination, the input interface is an IP interface, the output interface is one of an L2 interface and an IP interface, and the information database is a forwarding information base.

12. (withdrawn) The method according to claim 9, wherein:

for IP over MPLS tunnel origination, the input interface is an IP interface, the output interface is an L2 interface, and the information database is a forwarding information base.

13. (withdrawn) The method according to claim 9, wherein:

for L2TP tunnel origination, the input interface is a PPP interface, the output interface is one of an L2 interface and an IP interface, and the information database is a tunnel and session information base.

14. (previously presented) The method according to claim 9, wherein:

for ETHERNET over MPLS (multiprotocol label switching) tunnel origination, the input interface is an ETHERNET interface, the output interface is an L2 (layer 2) interface, and the information database is a switching information base.

15. (currently amended) A uniform method for implementing multiple tunneling protocols in a switch or router having a plurality of input streams and a plurality of output streams, comprising:

a) providing a finite set of tunnel interfaces, a first of said tunnel interfaces being associated with one tunneling protocol and a second of said tunnel interfaces being associated with a second tunneling protocol different from said first tunneling protocol; and

b) mapping input streams and output streams to tunnel interfaces for different tunneling protocols in a uniform manner.

16. (previously presented) The method according to claim 15, wherein:

some of the input streams are L2 (layer two) streams and some of the input streams are L3 (layer 3) streams, said step of providing a finite set of tunnel interfaces includes providing a set of L2 tunnel interfaces for L2 input streams and a set of L3 tunnel interfaces for L3 input streams.

17. (original) The method according to claim 16, wherein:

input streams are mapped to tunnel interfaces by a forwarding function.

18. (original) The method according to claim 16, wherein:

L2 input streams are mapped to L2 tunnel interfaces by a first forwarding function, and L3 input streams are mapped to L3 tunnel interfaces by a second forwarding function.

19. (original) The method according to claim 18, wherein:

some of the output streams are L2 streams and some of the output streams are L3 streams, L2 tunnel interfaces are mapped to L2 output streams by a third forwarding function, and L3 tunnel interfaces are mapped to L3 output streams by a fourth forwarding function.

20. (original) The method according to claim 19, wherein:

L2 tunnel interfaces are mapped to L3 output streams by a fifth forwarding function, and L3 tunnel interfaces are mapped to L2 output streams by a sixth forwarding function.

21. (previously presented) The method according to claim 17, wherein:

the forwarding function performs mapping based on context data associated with input packets and database information which is configured and updated by a local host.

22. (currently amended) A uniform method for implementing multiple tunneling protocols in a switch or router, comprising:

provitding a plurality of tunnel interfaces, a first of said tunnel interfaces being associated with one tunneling protocol and a second of said tunnel interfaces being associated with a second tunneling protocol different from said first tunneling protocol, each tunnel interface having a plurality of parameters which are described in a uniform way, said plurality of parameters including a local source address and a remote destination address.

23. (original) The method according to claim 22, wherein:

said plurality of parameters includes a hop limit or time to live.

24. (previously presented) The method according to claim 23, wherein:

said plurality of parameters includes a tunnel MTU (maximum transmission unit).

25. (original) The method according to claim 22, further comprising:

providing a plurality of tunnel entry node structures and a plurality of tunnel exit node structures

26. (original) The method according to claim 22, further comprising:

providing an address function to set tunnel interface source and destination addresses.

27. (previously presented) The method according to claim 26, further comprising:

provising a first address function for IPv4 (internet protocol version four) interfaces and a second address function for IPv6 (internet protocol version six) interfaces.

28. (original) The method according to claim 23, further comprising:

providing a hop function to set the hop limit for a tunnel interface.

29. (previously presented) The method according to claim 22, wherein:

said plurality of parameters includes MPLS (multiprotocol label switching) encapsulation information and actions to be performed on MPLS packets.

30. (previously presented) The method according to claim 29, further comprising:

providing an MPLS function to associate an MPLS LIB (label information base) with an MPLS interface.

31. (original) An application programming interface (API) for implementing a plurality of different tunneling protocols in a switch or router, said API comprising:

a) a tunneling interface data structure having a plurality of parameters; and

b) a plurality of functions for setting the parameters of the tunneling interface data structure, wherein

a tunneling interface data structure is configurable to implement any one of said plurality of different tunneling protocols by using at least some of said plurality of functions.

32. (original) The API according to claim 31, wherein:

said plurality of parameters including a local source address and a remote destination address.

33. (original) The API according to claim 32, wherein:

said plurality of parameters includes a hop limit or time to live.

34. (previously presented) The API according to claim 33, wherein:

said plurality of parameters includes a tunnel MTU (maximum transmission unit).

35. (original) The API according to claim 31, further comprising:

c) a plurality of tunnel entry node structures; and

d) a plurality of tunnel exit node structures.

36. (original) The API according to claim 31, wherein:

said plurality of functions includes an address function to set tunnel interface source and destination addresses.

37. (previously presented) The API according to claim 36, wherein:

said plurality of functions includes a first address function for IPv4 (internet protocol version four) interfaces and a second address function for IPv6 (internet protocol version six) interfaces.

38. (original) The API according to claim 33, wherein:

said plurality of functions includes a hop function to set the hop limit for a tunnel interface.

39. (previously presented) The API according to claim 31, wherein:

said plurality of parameters includes MPLS (multiprotocol label switching) encapsulation information and actions to be performed on MPLS packets.

40. (previously presented) The API according to claim 39, wherein:

said plurality of functions includes an MPLS function to associate an MPLS LIB (label information base) with an MPLS interface.